University
POLITEHNICA
of Bucharest

Faculty of
Automatic Control
and Computers

Computer Science
and Engineering
Department

Diagram Generation Using Genetic Algorithms and Orthogonal Routing

Bachelor Thesis – September 2014

Author

Andrei-Alexandru Mușat

Scientific Advisor(s)

As.Drd.Ing. Andrei Voinescu
As.Drd.Ing. Dan Dragomir

Introduction

- Diagrams are used as a tool to represent data and concepts
- Frequently utilised in software and hardware development
- Computing a proper drawing is a difficult task

Problem

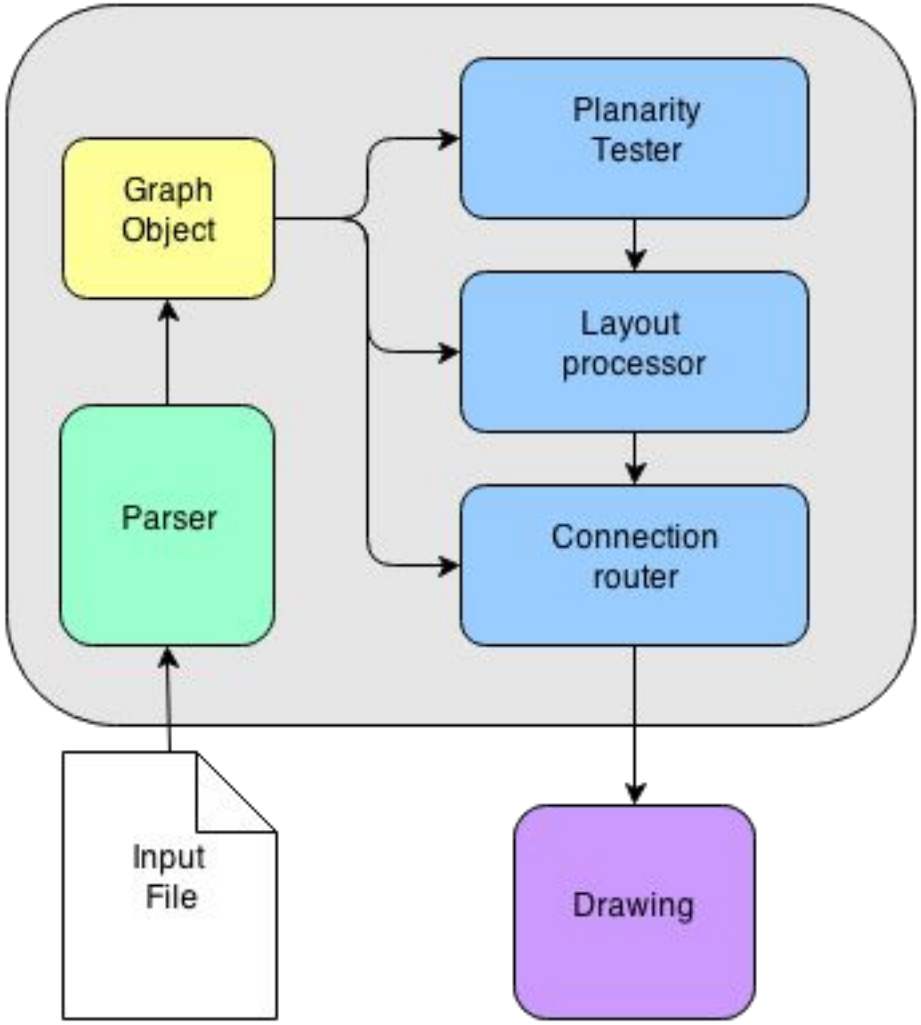
- NP-complete
- Classic algorithms have limitations
- Available software presents various drawbacks
- Difficult to satisfy every user request

Task Description

- Embed a graph in the plane
- Ensure planarity, clarity, orthogonality
- Optimize performance
- Allow user customization

System Architecture

- Four main modules
- Parser computes graph data from input files
- Planarity tester using Chiba-Nishizieki algorithm
- Layout processor based on genetic algorithms
- Edge router with orthogonal connectors



Implementation

- Modules implemented using Java language
- Eclipse Draw 2D used as graphical library
- Application integrated with the Eclipse IDE

Planarity

- Planar graphs have no intersecting edges
- Kuratowski's Theorem defines planar graphs
- Constant time algorithm tests necessary condition
- Booth-Lueker testing algorithm which runs in linear time

Graph layout

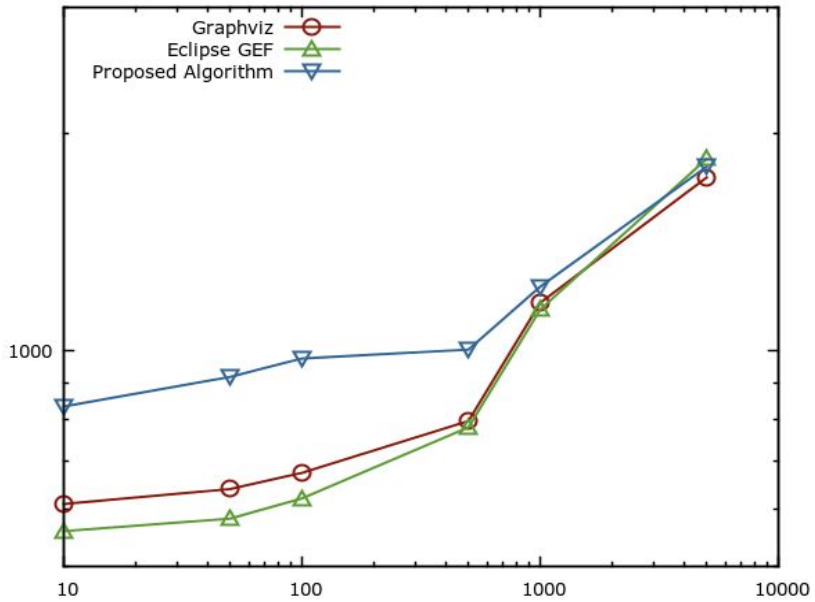
- Traditional methods: grid layout, tree layout, force-directed placing
- Experimental approach combines ideas
- Utilizes genetic algorithms
- Result is more clear and straightforward

Edge Routing

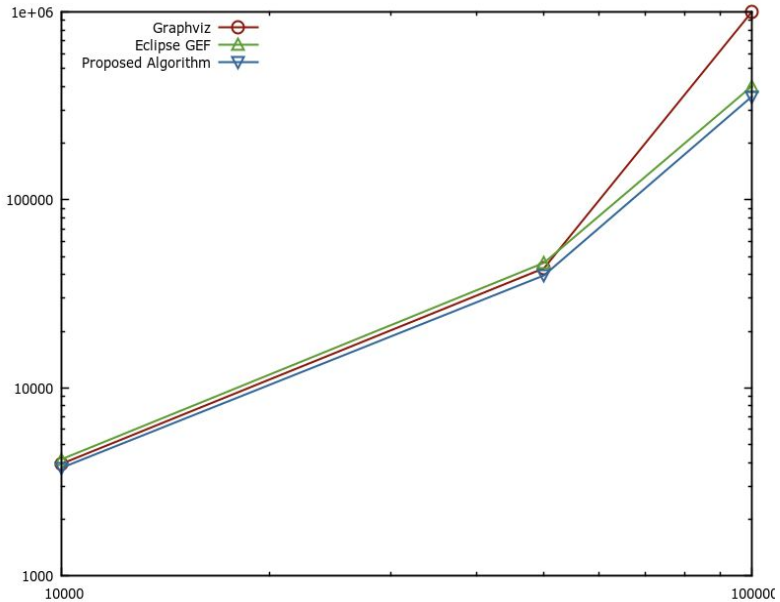
- Paths represented by orthogonal connectors
- Ensures clarity but occupies more space
- Advantageous for properly layed out graphs

Results - Benchmarking

- Benchmarking for main processing modules
- Available open-source software: Graphviz dot and Eclipse Gef used as reference



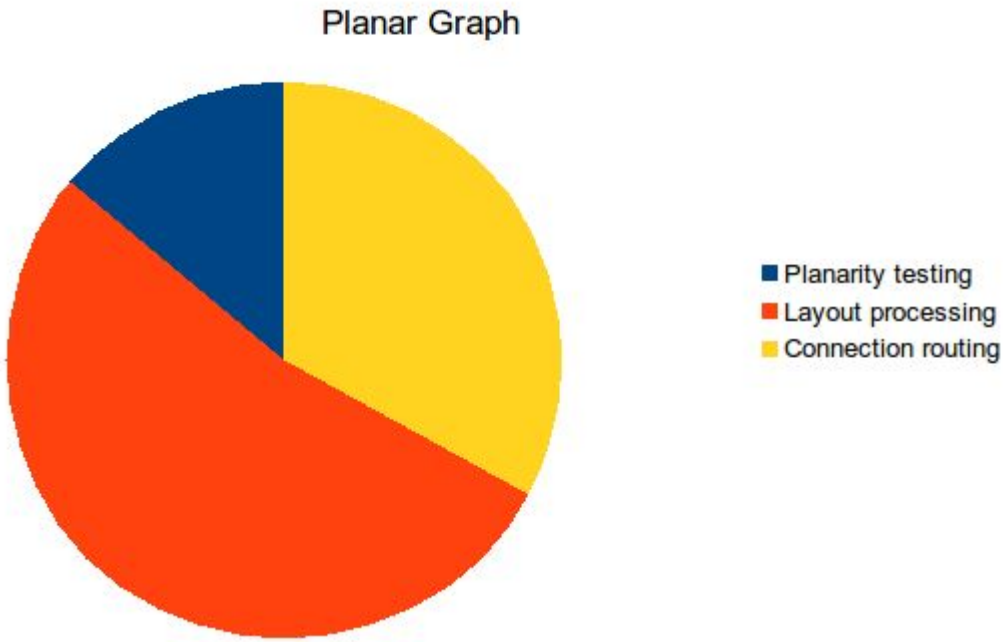
Benchmarking for graphs under 10000 nodes in logrithmic scale



Benchmarking for graphs over 10000 nodes in logrithmic scale

Performance Testing

- Identify intensively used module
- Possibility for optimization



User Oriented Features

- Interaction with final drawing
- Posibility of adjusting position of nodes
- On the fly routing for obstructed paths
- Pin elements

Conclusions

- Modern approach to a classic problem
- Using genetic algorithms improves performance
- Modularisation favours optimization
- Possibility of manual user adjustments of computed diagram provide insight for further improvements
- This work was accepted and presented at the 13th RoEduNet International Conference in Chișinău, Republic of Moldova